(3)

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/730,190	12/05/2000	Kestutis Patiejunas	MS160309.1	7993

27195 7590 10/19/2004

AMIN & TUROCY, LLP
24TH FLOOR, NATIONAL CITY CENTER
1900 EAST NINTH STREET
CLEVELAND, OH 44114

EXAMINER	
ALI, SYED J	
ART UNIT	PAPER NUMBER
2127	

DATE MAILED: 10/19/2004

Please find below and/or attached an Office communication concerning this application or proceeding.

PTO-90C (Rev. 10/03)

[illegible]

Office Action Summary	**Application No.** 09/730,190	**Applicant(s)** PATIEJUNAS, KESTUTIS	
	Examiner Syed J Ali	**Art Unit** 2127	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *20 July 2004*.

2a)☒ This action is **FINAL**. 2b)☐ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-50* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-50* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)	4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____ .
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)	5) ☐ Notice of Informal Patent Application (PTO-152)
3) ☐ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08) Paper No(s)/Mail Date _____.	6) ☐ Other: _____.

DETAILED ACTION

1. This office action is in response to the amendment filed July 20, 2004. Claims 1-50 are presented for examination.

2. The text of those sections of Title 35, U.S. code not included in this office action can be found in a prior office action.

Claim Rejections - 35 USC § 102

3. **Claim 1 is rejected under 35 U.S.C. 102(e) as being anticipated by Sievert et al. (USPN 6,687,729) (hereinafter Sievert).**

4. As per claim 1, Sievert teaches the invention as claimed, including a client side HTTP stack software component for processing requests, comprising:

at least one completion port object (col. 3 lines 20-32);

a thread pool comprising a plurality of threads adapted to process tasks associated with at least one client side request (col. 3 lines 20-32); and

a client side state machine associated with the at least one request (col. 3 lines 34-65).

5. **Claims 8, 23, 35, and 46 are rejected under 35 U.S.C. 102(b) as being anticipated by IBM Technical Disclosure Bulletin ("Control of Dynamic Threads Pool for Concurrent Remote Procedure Calls") (hereinafter IBM).**

6. As per claim 8, IBM teaches the invention as claimed, including a software component for implementing a client side HTTP stack, comprising:

a thread pool comprising N threads adapted to process M requests from a client application component, wherein N and M are integers greater than 1 and wherein M is greater than N (pg. 199).

7. As per claim 23, IBM teaches the invention as claimed, including a method of implementing a client side HTTP stack, comprising:

processing M requests from a client application component using a thread pool comprising N threads, wherein M and N are integers greater than 1 and wherein M is greater than N (pg. 199).

8. As per claim 35, IBM teaches the invention as claimed, including a computer-readable medium having computer-executable instructions for processing M requests from a client application component using a thread pool comprising N threads, wherein M and N are integers greater than 1 and wherein M is greater than N (pg. 199).

9. As per claim 46, IBM teaches the invention as claimed, including a software component for implementing a client side HTTP stack, comprising:

means for processing M requests from a client application component using a thread pool comprising N threads, wherein M and N are integers greater than 1 and wherein M is greater than N (pg. 199).

Claim Rejections - 35 USC § 103

10. **Claim 2 is rejected under 35 U.S.C. 103(a) as being unpatentable over Sievert in view of Jones et al. (USPN 6,003,061) (hereinafter Jones).**

11. As per claim 2, Jones teaches the invention as claimed, including the client side HTTP stack implementation of claim 1, further comprising a scheduler thread adapted to activate an object scheduled to begin sending requests at a specific time (col. 19 lines 39-49; col. 20 line 62 - col. 21 line 6).

12. It would have been obvious to one of ordinary skill in the art to combine Sievert and Jones since the prescheduling of threads allows the resource usage of a system to be known at compile time rather than run time. Particular advantages can be achieved in terms of load balancing and resource utilization by providing particular information related to the start time of an operation in advance. Additionally, the setting of a particular start time is beneficial to real time systems that have threads with hard deadlines or other scheduling constraints.

13. **Claim 3 is rejected under 35 U.S.C. 103(a) as being unpatentable over Sievert in view of Okano et al. (USPN 6,725,253) (hereinafter Okano).**

14. As per claim 3, Okano teaches the invention as claimed, including the client side HTTP stack implementation of claim 1, further comprising a DNS thread adapted to resolve domain names into IP addresses (col. 12 line 37 - col. 13 line 5).

15. It would have been obvious to one of ordinary skill in the art to combine Sievert and Okano since IP addresses are expressed in octets that make it difficult to remember domain names. Rather, easy to remember domain names are provided that are then translated into IP addresses easing the use of a networked system by a user (Okano, col. 2 lines 4-10).

16. **Claim 4 is rejected under 35 U.S.C. 103(a) as being unpatentable over Sievert in view of Paxhia et al. (USPN 6,493,749) (hereinafter Paxhia).**

17. As per claim 4, Paxhia teaches the invention as claimed, including the client side HTTP stack implementation of claim 1, further comprising a timeout thread with a list of active sockets and timers associated with each socket, and adapted to selectively timeout at least one socket according to at least one timer in the list (col. 41 lines 19-28).

18. It would have been obvious to one of ordinary skill in the art to combine Sievert and Paxhia since a thread that has been operating for an extended period of time without responding may be causing a starvation condition. The use of a timer to monitor a socket ensures that a thread does not stall while utilizing one of the system's sockets. The expiration of the timer thus alarms the system that the thread should be terminated, thereby protecting system resources and ensuring that other threads receive a fair share of the processor.

19. **Claim 5 is rejected under 35 U.S.C. 103(a) as being unpatentable over Sievert in view of Paxhia as applied to claim 4 above, and further in view of Jones.**

20. As per claim 5, Jones teaches the invention as claimed, including the client side HTTP stack implementation of claim 4, farther comprising a scheduler thread adapted to activate an object scheduled to begin sending requests at a specific time (col. 19 lines 39-49; col. 20 line 62 - col. 21 line 6).

21. It would have been obvious to one of ordinary skill in the art to combine Sievert, Paxhia, and Jones since the prescheduling of threads allows the resource usage of a system to be known at compile time rather than run time. Particular advantages can be achieved in terms of load balancing and resource utilization by providing particular information related to the start time of an operation in advance. Additionally, the setting of a particular start time is beneficial to real time systems that have threads with hard deadlines or other scheduling constraints.

22. **Claim 6 is rejected under 35 U.S.C. 103(a) as being unpatentable over Sievert in view of Paxhia in view of Jones as applied to claim 5 above, and further in view of Okano.**

23. As per claim 6, Okano teaches the invention as claimed, including the client side HTTP stack implementation of claim 5, further comprising a DNS thread adapted to resolve domain names into IP addresses (col. 12 line 37 - col. 13 line 5).

24. It would have been obvious to one of ordinary skill in the art to combine Sievert, Paxhia, Jones, and Okano since IP addresses are expressed in octets that make it difficult to remember domain names. Rather, easy to remember domain names are provided that are then translated into IP addresses easing the use of a networked system by a user (Okano, col. 2 lines 4-10).

25. **Claim 7 is rejected under 35 U.S.C. 103(a) as being unpatentable over Sievert in view of Paxhia as applied to claim 4 above, and further in view of Okano.**

26. As per claim 7, Okano teaches the invention as claimed, including the client side HTTP stack implementation of claim 4, further comprising a DNS thread adapted to resolve domain names into IP addresses (col. 12 line 37 - col. 13 line 5).

27. It would have been obvious to one of ordinary skill in the art to combine Sievert, Paxhia, and Okano since IP addresses are expressed in octets that make it difficult to remember domain names. Rather, easy to remember domain names are provided that are then translated into IP addresses easing the use of a networked system by a user (Okano, col. 2 lines 4-10).

28. **Claims 9-13, 17-19, 24-28, 32-34, 36-39, and 47 are rejected under 35 U.S.C. 103(a) as being unpatentable over IBM in view of Sievert.**

29. As per claim 9, Sievert teaches the invention as claimed, including the software component of claim 8, further comprising at least one thread activation component adapted to activate at least one of the N threads based on an event (col. 3 lines 45-52).

30. It would have been obvious to one of ordinary skill in the art to combine IBM and Sievert since the method of IBM is absent guidance as to how threads are handled in terms of sending and receiving data. IBM is limited to showing a method for initializing and controlling the size of a thread pool. Sievert provides additional functionality for a pool of threads to handle work requests as well as encapsulating requests and responses within an I/O completion port, thereby

easing the manner in which requests are handled. The use of a completion port is beneficial in that it simplifies distributed computing for multiple concurrent requests by handling all incoming and outgoing data.

31. As per claim 10, Sievert teaches the invention as claimed, including the software component of claim 9, wherein the at least one thread activation component is a completion port (col. 3 lines 20-32).

32. As per claim 11, Sievert teaches the invention as claimed, including the software component of claim 9, wherein at least one of the N threads is adapted to deactivate itself and return to the thread pool when an operation being processed by the at least one of the threads is pending (col. 5 lines 26-38).

33. As per claim 12, Sievert teaches the invention as claimed, including the software component of claim 11, wherein the event is the receipt of a completion packet by the at least one thread activation component (col. 3 lines 27-32).

34. As per claim 13, Sievert teaches the invention as claimed, including the software component of claim 12, wherein the at least one thread activation component is a completion port (col. 3 lines 20-32).

35. As per claim 17, Sievert teaches the invention as claimed, including the software component of claim 9, further comprising a state machine associated with at least one of the M requests (col. 3 lines 34-65).

36. As per claim 18, Sievert teaches the invention as claimed, including the software component of claim 17, further comprising at least one key associated with the at least one of the M requests, wherein a first one of the N threads is associated with the at least one of the M requests, and wherein the thread activation component is adapted to associate the context of the first one of the N threads with the at least one state machine using the at least one key, in order to activate the first one of the N threads (col. 5 line 59 - col. 6 line 54).

37. As per claim 19, Sievert teaches the invention as claimed, including the software component of claim 18, wherein the thread activation component is adapted to associate the context of one of the N threads with the at least one state machine using the at least one key in order to activate the one of the N threads based on an event (col. 5 line 59 - col. 6 line 54).

38. As per claim 24, Sievert teaches the invention as claimed, including the method of claim 23, further comprising:

selectively deactivating at least one of the N threads (col. 5 lines 26-38); and

activating at least another of the N threads based on an event using at least one thread activation component (col. 3 lines 45-52).

39. As per claim 25, Sievert teaches the invention as claimed, including the method of claim 24, wherein the at least one thread activation component is a completion port (col. 3 lines 20-32).

40. As per claim 26, Sievert teaches the invention as claimed, including the method of claim 24, wherein selectively deactivating at least one of the N threads comprises deactivating the at least one of the N threads when an operation being processed by the at least one of the N threads is pending (col. 5 lines 26-38).

41. As per claim 27, Sievert teaches the invention as claimed, including the method of claim 26, wherein activating at least another of the N threads based on an event comprises:

receiving a completion packet using the thread activation component (col. 3 lines 27-32); and

activating one of the N threads upon receipt of the completion packet using the thread activation component (col. 3 lines 45-52).

42. As per claim 28, Sievert teaches the invention as claimed, including the method of claim 27, wherein the at least one thread activation component is a completion port (col. 3 lines 20-32).

43. As per claim 32, Sievert teaches the invention as claimed, including the method of claim 26, further comprising associating a state machine with at least one of the M requests (col. 3 lines 34-65).

44. As per claim 33, Sievert teaches the invention as claimed, including the method of claim 32, further comprising:

associating at least one key with the at least one of the M requests (col. 5 line 59 - col. 6 line 54);

associating a first one of the N threads with the at least one of the M requests (col. 5 line 59 - col. 6 line 54); and

associating a context of the first one of the N threads with the at least one state machine using the at least one key, in order to deactivate the first one of the N threads (col. 5 lines 26-38; col. 5 line 59 - col. 6 line 54).

45. As per claim 34, Sievert teaches the invention as claimed, including the method of claim 33, further comprising associating a context of one of the N threads with the at least one state machine using the at least one key in order to activate the one of the N threads based on an event (col. 5 line 59 - col. 6 line 54).

46. As per claim 36, Sievert teaches the invention as claimed, including the computer-readable medium of claim 35, further comprising computer-executable instructions for:

selectively deactivating at least one of the N threads (col. 5 lines 26-38); and

activating at least another of the N threads based on an event using at least one thread activation component (col. 3 lines 45-52).

47. As per claim 37, Sievert teaches the invention as claimed, including the computer-readable medium of claim 36, wherein the at least one thread activation component is a completion port (col. 3 lines 20-32).

48. As per claim 38, Sievert teaches the invention as claimed, including the computer-readable medium of claim 36, wherein the computer-executable instructions for selectively deactivating at least one of the N threads comprises computer-executable instructions for deactivating the at least one of the N threads when an operation being processed by the at least one of the N threads is pending (col. 5 lines 26-38).

49. As per claim 39, Sievert teaches the invention as claimed, including the computer-readable medium of claim 38, wherein the computer-executable instructions for activating at least another of the N threads based on an event comprises computer-executable instructions for:

receiving a completion packet using the thread activation component (col. 3 lines 27-32); and

activating one of the N threads upon receipt of the completion packet using the thread activation component (col. 3 lines 45-52).

50. As per claim 47, Sievert teaches the invention as claimed, including the software component of claim 46, further comprising:

means for selectively deactivating at least one of the N threads (col. 5 lines 26-38); and means for activating at least another of the N threads based on an event (col. 3 lines 45-52).

51. **Claims 14, 29, 40, and 48 are rejected under 35 U.S.C. 103(a) as being unpatentable over IBM in view of Sievert as applied to claims 13, 28, 39, and 47 above respectively, and further in view of Jones.**

52. As per claim 14, Jones teaches the invention as claimed, including the software component of claim 13, further comprising a scheduler thread adapted to activate an object scheduled to begin sending requests at a specific time (col. 19 lines 39-49; col. 20 line 62 - col. 21 line 6).

53. It would have been obvious to one of ordinary skill in the art to combine IBM, Sievert, and Jones since the prescheduling of threads allows the resource usage of a system to be known at compile time rather than run time. Particular advantages can be achieved in terms of load balancing and resource utilization by providing particular information related to the start time of an operation in advance. Additionally, the setting of a particular start time is beneficial to real time systems that have threads with hard deadlines or other scheduling constraints.

54. As per claim 29, Jones teaches the invention as claimed, including the method of claim 28, further comprising activating an object scheduled to begin sending requests at a specific time using a scheduler thread (col. 19 lines 39-49; col. 20 line 62 - col. 21 line 6).

55. As per claim 40, Jones teaches the invention as claimed, including the computer-readable medium of claim 39, further comprising computer-executable instructions for activating an object scheduled to begin sending requests at a specific time using a scheduler thread (col. 19 lines 39-49; col. 20 line 62 - col. 21 line 6).

56. As per claim 48, Jones teaches the invention as claimed, including the software component of claim 47, further comprising means for activating an object scheduled to begin sending requests at a specific time (col. 19 lines 39-49; col. 20 line 62 - col. 21 line 6).

57. **Claims 15, 30, 41 are rejected under 35 U.S.C. 103(a) as being unpatentable over IBM in view of Sievert in view of Jones as applied to claims 14, 29, and 40 above respectively, and further in view of Okano.**

58. As per claim 15, Okano teaches the invention as claimed, including the software component of claim 14, further comprising a DNS thread adapted to resolve domain names into IP addresses (col. 12 line 37 - col. 13 line 5).

59. It would have been obvious to one of ordinary skill in the art to combine IBM, Sievert, Jones, and Okano since IP addresses are expressed in octets that make it difficult to remember domain names. Rather, easy to remember domain names are provided that are then translated into IP addresses easing the use of a networked system by a user (Okano, col. 2 lines 4-10).

60. As per claim 30, Okano teaches the invention as claimed, including the method of claim 29, further comprising resolving domain names into IP addresses using a DNS thread (col. 12 line 37 - col. 13 line 5).

61. As per claim 41, Okano teaches the invention as claimed, including the computer-readable medium of claim 40, further comprising computer-executable instructions for resolving domain names into IP addresses using a DNS thread (col. 12 line 37 - col. 13 line 5).

62. **Claims 16, 31, and 42-45 are rejected under 35 U.S.C. 103(a) as being unpatentable over IBM in view of Sievert in view of Jones in view of Okano as applied to claims 15, 30, and 41 above respectively, and further in view of Paxhia.**

63. As per claim 16, Paxhia teaches the invention as claimed, including the software component of claim 15, further comprising a timeout thread with a list of active sockets and timers associated with each socket, and adapted to selectively timeout at least one socket according to at least one timer in the list (col. 41 lines 19-28).

It would have been obvious to one of ordinary skill in the art to combine IBM, Sievert, Jones, Okano, and Paxhia since a thread that has been operating for an extended period of time without responding may be causing a starvation condition. The use of a timer to monitor a socket ensures that a thread does not stall while utilizing one of the system's sockets. The expiration of the timer thus alarms the system that the thread should be terminated, thereby protecting system resources and ensuring that other threads receive a fair share of the processor.

64. As per claim 31, Paxhia teaches the invention as claimed, including the method of claim 30, further comprising selectively timing out at least one socket according to at least one timer associated with the at least one socket using a timeout thread comprising a list of active sockets and timers associated with each socket (col. 41 lines 19-28).

65. As per claim 42, Paxhia teaches the invention as claimed, including the computer-readable medium of claim 41, further comprising computer-executable instructions for selectively timing out at least one socket according to at least one timer associated with the at least one socket using a timeout thread comprising a list of active sockets and timers associated with each socket (col. 41 lines 19-28).

66. As per claim 43, Sievert teaches the invention as claimed, including the computer-readable medium of claim 42, further comprising computer-executable instructions for associating a state machine with at least one of the M requests (col. 3 lines 34-65).

67. As per claim 44, Sievert teaches the invention as claimed, including the computer-readable medium of claim 43, further comprising computer-executable instructions for:

associating at least one key with the at least one of the M requests (col. 5 line 59 - col. 6 line 54);

associating a first one of the N threads with the at least one of the M requests (col. 5 line 59 - col. 6 line 54); and associating a context of the first one of the N threads with the at least one state machine using the at least one key, in order to deactivate the first one of the N threads (col. 5 line 59 - col. 6 line 54).

68. As per claim 45, Sievert teaches the invention as claimed, including the computer-readable medium of claim 44, further comprising computer-executable instructions for associating a context of one of the N threads with the at least one state machine using the at least one key in order to activate the one of the N threads based on an event (col. 5 line 59 - col. 6 line 54).

69. **Claim 20 is rejected under 35 U.S.C. 103(a) as being unpatentable over IBM in view of Jones.**

70. As per claim 20, Jones teaches the invention as claimed, including the software component of claim 8, further comprising a scheduler thread adapted to activate an object scheduled to begin sending requests at a specific time (col. 19 lines 39-49; col. 20 line 62 - col. 21 line 6).

71. It would have been obvious to one of ordinary skill in the art to combine IBM and Jones since the prescheduling of threads allows the resource usage of a system to be known at compile time rather than run time. Particular advantages can be achieved in terms of load balancing and resource utilization by providing particular information related to the start time of an operation in

advance. Additionally, the setting of a particular start time is beneficial to real time systems that have threads with hard deadlines or other scheduling constraints.

72. **Claim 21 is rejected under 35 U.S.C. 103(a) as being unpatentable over IBM in view of Okano.**

73. As per claim 21, Okano teaches the invention as claimed, including the software component of claim 8, further comprising a DNS thread adapted to resolve domain names into IP addresses (col. 12 line 37 - col. 13 line 5).

74. It would have been obvious to one of ordinary skill in the art to combine IBM and Okano since IP addresses are expressed in octets that make it difficult to remember domain names. Rather, easy to remember domain names are provided that are then translated into IP addresses easing the use of a networked system by a user (Okano, col. 2 lines 4-10).

75. **Claim 22 is rejected under 35 U.S.C. 103(a) as being unpatentable over IBM in view of Paxhia.**

76. As per claim 22, Paxhia teaches the invention as claimed, including the software component of claim 8, further comprising a timeout thread with a list of active sockets and timers associated with each socket, and adapted to selectively timeout at least one socket according to at least one timer in the list (col. 41 lines 19-28).

77. It would have been obvious to one of ordinary skill in the art to combine IBM and Paxhia since a thread that has been operating for an extended period of time without responding may be causing a starvation condition. The use of a timer to monitor a socket ensures that a thread does not stall while utilizing one of the system's sockets. The expiration of the timer thus alarms the system that the thread should be terminated, thereby protecting system resources and ensuring that other threads receive a fair share of the processor.

78. **Claim 49 is rejected under 35 U.S.C. 103(a) as being unpatentable over IBM in view of Sievert as applied to claim 47 above, and further in view of Okano.**

79. As per claim 49, Okano teaches the invention as claimed, including the software component of claim 47, further comprising means for resolving domain names into IP addresses (col. 12 line 37 - col. 13 line 5).

80. It would have been obvious to one of ordinary skill in the art to combine IBM, Sievert, and Okano since IP addresses are expressed in octets that make it difficult to remember domain names. Rather, easy to remember domain names are provided that are then translated into IP addresses easing the use of a networked system by a user (Okano, col. 2 lines 4-10).

81. **Claim 50 is rejected under 35 U.S.C. 103(a) as being unpatentable over IBM in view of Sievert as applied to claim 47 above, and further in view of Paxhia.**

82. As per claim 50, Paxhia teaches the invention as claimed, including the software component of claim 47, further comprising means for selectively timing out at least one socket according to at least one timer associated with the at least one socket (col. 41 lines 19-28).

83. It would have been obvious to one of ordinary skill in the art to combine IBM and Paxhia since a thread that has been operating for an extended period of time without responding may be causing a starvation condition. The use of a timer to monitor a socket ensures that a thread does not stall while utilizing one of the system's sockets. The expiration of the timer thus alarms the system that the thread should be terminated, thereby protecting system resources and ensuring that other threads receive a fair share of the processor.

Response to Arguments

84. Applicant's arguments filed July 20, 2004 have been fully considered but they are not persuasive.

85. Applicant presents the following arguments regarding Sievert:

"*The [completion] port mentioned in Sievert et al. utilizes a physical port through which data is processed, while the claimed invention proposes an object within a program to further functionality of the program.*"

"*Sievert et al. proposes a thread pool for 'performing items of work'...that does not have any obvious applications associated with the processing of at least* ***one client side request.***"

"*Sievert et al. discloses a state machine that...does not have any obvious applications to a state machine that is related to, or associated with, an HTTP request as recited in claim 1.*"

"[T]he thread pool manager recited in Sievert et al. fails to mention any sort of completion packet."

"[T]he 'state machine' in the claimed invention differs from that of the cited document, in that the claimed invention utilizes a state machine that functions with respect to individual threads, while the document cited requires the state machine to function on a work queue data structure."

86. Applicant has erroneously characterized the I/O completion port as a physical port. Sievert discusses the use of the Windows NT operating system I/O completion port, which is known to be implemented as a software object. Rowe (USPN 6,324,492) discusses that Windows NT completion ports are implemented as software objects (col. 4 lines 24-42).

In response to Applicant's argument that the thread pool and state machine discussed by Sievert is deficient for not being related to HTTP requests, a recitation of the intended use of the claimed invention must result in a structural difference between the claimed invention and the prior art in order to patentably distinguish the claimed invention from the prior art. If the prior art structure is capable of performing the intended use, then it meets the claim. In a claim drawn to a process of making, the intended use must result in a manipulative difference as compared to the prior art. See *In re Casey*, 152 USPQ 235 (CCPA 1967) and *In re Otto*, 136 USPQ 458, 459 (CCPA 1963).

In response to Applicant's argument that Sievert does not discuss a completion packet, Examiner respectfully disagrees. Packets are used generally to refer to a unit of data at any layer of the OSI protocol stack. Packets are typically subject to encapsulation, where each component that performs an operation upon the data in a packet either adds header information or strips header information to further the processing. In Sievert, API calls are wrapped by thread operations, and later provided to the I/O completion port when processing has completed. This method of processing meets the basic requirements of what purpose packets serve.

Finally, Examiner respectfully disagrees that Sievert fails to utilize a state machine that functions with respect to individual threads. While Sievert does in fact use the state machine in conjunction with a work queue data structure, the work queue is the data structure by which individual threads are serviced and perform work. There is no limitation in the claims that indicate that there can be no intervening data structures to aid with the processing of requests.

87. Applicant presents the following arguments regarding IBM:

"[T]he method disclosed by this reference is applicable to an application server receiving Remote Procedure Calls from other entities rather than a client device making HTTP requests."

"The algorithm defined in the reference functions on executor threads, which are utilized to facilitate the functionality of an application server; however, such threads are not of relevance in a ***client-side*** *implementation designed to effectively handle requests."*

88. IBM discusses managing the thread pool for managing requests made by an application server. While the remote server may typically handle RPC calls, there is no reason to believe that the thread pool could not be implemented on the client side. Each thread is created to perform some sort of work, wherein the method of managing the thread pool could be implemented to serve any number of computing functions that utilize multithreading. Additionally, recitation of the intended use of the claimed invention must result in a structural difference between the claimed invention and the prior art in order to patentably distinguish the

claimed invention from the prior art. If the prior art structure is capable of performing the intended use, then it meets the claim. In a claim drawn to a process of making, the intended use must result in a manipulative difference as compared to the prior art. See *In re Casey*, 152 USPQ 235 (CCPA 1967) and *In re Otto*, 136 USPQ 458, 459 (CCPA 1963). In this particular case, the thread pool of IBM could be implemented on the client side easily if the developer so chose.

89. Applicant presents the following arguments regarding Jones:

"Jones et al. discloses a 'thread scheduling facility'...to manage thread operations, however this object is not defined to function as a thread itself."

"[T]he scheduler implemented in Jones et al. creates 'thread data structure[s]'...instead of activating an object as recited."

90. Jones discusses that the thread scheduling mechanism is implemented as a software facility. The specifics of how the software facility is designed are highly dependent upon the type of system that is being used. For instance, if the system is one that makes extensive use of multithreading, including implementation of the operating system as a thread, then it would follow that the scheduling facility would be implemented as a thread. While it is noted that Jones does not explicitly state the thread scheduling facility is implemented as a thread, this feature is certainly supported, and would be a design choice. Additionally, the language of "activating" an object is not patentably distinct from creating a thread object. When the thread is created to perform work, it is automatically activated to perform some processing. As the thread

creation and deletion facilities operate dynamically, the scheduler and thread pool managers activate or create threads as needed.

91. Applicant presents the following arguments regarding Okano:

"*The claimed invention differs most notably from [Okano] because it defines a DNS thread that is using information from a queue data structure to perform the resolution and signal an event upon completion…instead of having some other object or process perform the name-to-IP address translation.*"

92. Examiner respectfully disagrees, in that Okano specifically discusses the use of a DNS thread for performing various DNS operations such as sorting, monitoring, and distribution, in addition to name translation (col. 12 line 37 - col. 13 line 5).

93. Applicant presents the following arguments regarding Paxhia:

"*Paxhia et al. defines an 'alarm thread' that 'wakes up periodically and checks a list of timers' to determine the status of threads that are currently being processed (See Paxhia et al. col. 41). This 'alarm thread' does not selectively time out threads as defined.*"

94. Examiner respectfully disagrees, in that Paxhia specifically discusses the selective timing out of threads based on checking a list of timers to determine which have terminated processing or not completed processing in time.

Conclusion

95. **THIS ACTION IS MADE FINAL.** Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire THREE MONTHS from the mailing date of this action. In the event a first reply is filed within TWO MONTHS of the mailing date of this final action and the advisory action is not mailed until after the end of the THREE-MONTH shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event, however, will the statutory period for reply expire later than SIX MONTHS from the mailing date of this final action.

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Syed J Ali whose telephone number is (571) 272-3769. The examiner can normally be reached on Mon-Fri 8-5:30, 2nd Friday off.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Meng-Ai T An can be reached on (571) 272-3756. The fax phone number for the organization where this application or proceeding is assigned is 703-872-9306.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

[illegible]

Syed Ali
October 4, 2004

[illegible]
MENG-AI T. AN
SUPERVISORY PATENT EXAMINER
TECHNOLOGY CENTER 2100